Au cœur du Web participatif : Ajax

Travail de diplôme réalisé en vue de l'obtention du diplôme HES

par :

Alain PELLAUX

Conseiller au travail de diplôme :

Peter DAEHNE, Professeur HES

Genève, le 27 novembre 2007

Haute École de Gestion de Genève (HEG-GE)

Filière Informatique de Gestion

Déclaration

Ce travail de diplôme est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre de bachelor en informatique de gestion. L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de diplôme, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de diplôme, du juré et de la HEG.

« J'atteste avoir réalisé seul le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 27 novembre 2007

Alain PELLAUX

Remerciements

Je tiens à remercier M. Peter DAEHNE pour son suivi, sa disponibilité et ses précieux conseils tout au long de ce travail de diplôme.

Je remercie également le Centre Info de la HEG pour la mise à disposition d'un hébergement sur le site campus.

Sommaire

Ce travail de diplôme est l'accomplissement du dernier module nécessaire en vue de l'obtention du titre de Bachelor en Informatique de Gestion. Tout au long de la réalisation de ce travail de diplôme, nous avons été suivi par M. Peter Daehne, Professeur HES à la HEG.

L'objectif principal de ce mémoire fut l'étude des différentes technologies composant Ajax et la réalisation d'un prototype d'application illustrant leur mise en œuvre.

Pour ce faire nous avons commencé par exposer ce qu'est le Web et son évolution, tout en faisant le lien entre l'apparition d'applications Web se rapprochant de celles dites de bureau et l'apparition d'Ajax.

Par la suite, nous avons expliqué le principe de fonctionnement d'Ajax et ce qu'il apporte, puis détaillé les différentes technologies qui le composent. Nous avons présenté la syntaxe et le fonctionnement de ces différents langages, qui peuvent d'ailleurs tous fonctionner de manière indépendants les unes des autres.

Enfin, après avoir détaillé le prototype d'application et son fonctionnement, nous avons parlé, avec les frameworks, des différents moyens mis à disposition d'un développeur pour construire une application Ajax, de ses points forts et de ses points faibles ainsi que des autres méthodes permettant d'obtenir des résultats similaires à une application Ajax.

Table des matières

Liste des Tableaux

Liste des Figures

Liste des Illustrations

Introduction

Le monde de l'Internet est en constante évolution, il est donc normal que les méthodes de conception de sites Internet évoluent elles aussi.

Partant de ce principe et étant passionné par Internet et tout ce qui s'y rapporte nous avons décidé de réaliser ce travail de diplôme en étudiant l'une des technologies les plus utilisées dans le microcosme du Web 2.0 : Ajax.

Ajax est en effet très utilisé dans les sites communautaires et nous avait été présenté très brièvement lors d'un cours à la HEG. Désirant réaliser un mémoire contenant une partie pratique nous avons saisi cette occasion pour approfondir nos connaissances en la matière.

Ce travail est donc découpé en deux parties principales. La première expose les différentes technologies composant Ajax. La seconde consiste en une mise en pratique de ces technologies au travers d'une application permettant d'obtenir certaines informations sur les communes de Suisse romande à partir de leur nom saisi dans un formulaire.

Nous concluons cette étude en détaillant le rôle et le code des différents fichiers constituant l'application.

1. L'évolution du Web

1.1 Le Web 1.0

Le World Wide Web, plus souvent désigné par l'acronyme[1] WWW, appelé également Web est apparu au début des années 90 et représente les sites Internet dits statiques ; c'est ce que l'on appelle le Web 1.0.

Le contenu est composé de textes, d'images, de sons et de vidéos, liés par des liens hypertextes[2] et hébergés sur un serveur ; l'utilisateur peut y accéder à travers un navigateur Internet comme Internet Explorer de Microsoft depuis n'importe où dans le monde.

Les pages composant le Web 1.0 ne sont pas régulièrement mises à jour et une personne consultant une page ne pourra y apporter sa contribution autrement qu'en envoyant un e-mail à l'auteur.

L'apparition de nouveaux langages comme ASP[3] ou PHP[4], qui permettent de se connecter à une base de données, permet de faire évoluer le Web, les sites Internet et leur contenu qui devient dynamique, c'est ce que l'on pourrait appeler le Web 1.5.

1.2 Le Web 2.0

Aussi appelé Web participatif, le Web 2.0 est un concept qui est apparu pour la première fois en août 2004 lors d'une conférence « brainstorming »[5] entre deux sociétés, O'Reilly et Medialive International.

Aujourd'hui encore il est difficile de décrire exactement ce qu'est le Web 2.0, mais s'il n'est pas possible de donner une définition précise du terme il est en revanche possible d'en expliquer le sens à travers des exemples.

Un article publié par Tim O'Reilly[6] décrit des exemples de différences entre sites « 1.0 » et sites « 2.0 », issu du brainstorming cité plus haut. Parmi ceux-ci, deux sont

---

[1] Glossaire – [A]

[2] Glossaire – [H]

[3] Glossaire – [A]

[4] Glossaire – [P]

[5] Glossaire – [B]

[6] [web04]

particulièrement parlants, à savoir que des sites personnels du Web 1.0, le Web 2.0 a amené les blogs[7] et que les publications du Web 1.0 ont évolué en participation avec le Web 2.0. Ces deux exemples montrent bien que l'appellation Web participatif du Web 2.0 n'est pas usurpée. Ainsi YouTube[8], l'un des sites les plus populaires de la planète à l'heure actuelle, ne serait rien sans l'apport de ses membres, de même que l'encyclopédie en ligne Wikipedia[9].

Alors qu'avec le Web 1.0 l'internaute était passif, celui-ci devient actif avec le Web 2.0. Il peut ainsi se créer une page d'accueil personnalisée à l'aide de Netvibes[10] ou d'iGoogle[11], lui permettant ainsi de gérer ses tâches, consulter la météo, des flux RSS[12] et bien d'autres choses encore grâce aux modules[13] que l'on peut ajouter en quelques clics. Le tout dans une page entièrement personnalisable et ne nécessitant aucune compétence en informatique.

Des géants de l'Internet tels que Google ou plus récemment Microsoft proposent des services en ligne permettant de créer des documents, des feuilles de calcul ou encore des présentations. Ces fichiers sont ensuite stockés en ligne et permettent aux internautes de travailler simultanément sur des fichiers partagés. La multiplication de ces services, toujours plus proches d'une application de bureau traditionnelle, peut laisser imaginer à terme la disparition des logiciels de bureau tels que nous les connaissons aujourd'hui.

Un autre aspect important du Web 2.0 est l'aspect de communauté qui en ressort. Au travers des blogs tout d'abord, où les commentaires des lecteurs apportent souvent un plus non négligeable au texte de base, mais aussi aux travers de sites tels que MySpace[14] ou encore, pour celui que nous trouvons le plus intéressant, Facebook[15].

---

[7] Glossaire – [B]

[8] http://www.youtube.com/

[9] http://www.wikipedia.org/

[10] http://www.netvibes.com/

[11] http://www.google.com/ig

[12] Glossaire – [F]

[13] Glossaire – [M]

[14] http://www.myspace.com/

[15] http://www.facebook.com/

Facebook est un site communautaire ayant pour but de retrouver des amis inscrits eux aussi sur le site. À la base réservé aux seuls étudiants de Harvard, il est maintenant ouvert à tous.

Facebook permet donc de retrouver très simplement des personnes en entrant leur nom et/ou d'autres informations. En examinant le profil d'une personne on peut voir ses amis ainsi que ceux que l'on a éventuellement en commun et ainsi retrouver des personnes en quelques clics, puis reprendre contact avec elles via une messagerie publique ou privée. Il est aussi possible de se joindre à divers groupes, permettant ainsi aux utilisateurs de se créer un véritable réseau social.

L'intérêt de Facebook ne s'arrête pas là puisque le site permet, à la manière de Netvibes, de personnaliser la page de son profil mais aussi d'ajouter des applications très diverses telles que des jeux, des questionnaires et autres, puis de faire partager ses découvertes à ses amis.

Au travers de tous ces exemples, on peut donc dire que le Web 2.0 est une évolution du Web où chaque internaute est acteur d'un réseau social dont il partage les informations.

1.3 Le Web 3.0

Le Web 3.0 est aussi appelé Web sémantique. Il fait encore à l'heure actuelle l'objet de discussions au sein du consortium W3C.

Alors que le Web d'aujourd'hui permet de connecter des documents, le Web 3.0 sera lui un Web ou ce seront les données qui seront connectées ensemble.

John Markoff, journaliste au New York Times, indiquait ceci dans un de ses articles[16] :

> *In contrast, the Holy Grail for developers of the semantic Web is to build a system that can give a reasonable and complete response to a simple question like: "I'm looking for a warm place to vacation and I have a budget of $3,000. Oh, and I have an 11-year-old child."*
>
> *Under today's system, such a query can lead to hours of sifting — through lists of flights, hotel, car rentals — and the options are often at odds with one another. Under Web 3.0, the same search would ideally call up a complete vacation package that was planned as meticulously as if it had been assembled by a human travel agent.* *(John Markoff, 2006: 1)*

[16] [web03]

Soit que le Saint Graal pour les développeurs du Web sémantique serait de créer un système qui puisse donner une réponse raisonnable et complète à une question simple comme : « je cherche un endroit chaud pour les vacances et j'ai un budget de 3'000$. Oh, j'ai un enfant de 11 ans ».

John Markoff poursuit en expliquant qu'aujourd'hui, répondre à cette question pourrait conduire à des heures de recherche pour des avions, hôtels et locations de voitures tout en donnant des résultats contradictoires. Tandis qu'avec le Web 3.0, la même recherche donnerait comme réponse un forfait complet planifié aussi méticuleusement que le ferait un agent de voyage humain.

2. Ajax

2.1 Définition du terme Ajax

Le terme Ajax a été inventé par Jesse James Garrett et est apparu pour la première fois dans un article[17] publié le 18 février 2005 et intitulé « Ajax : A New Approach to Web Applications ». Garrett a expliqué par la suite avoir commencé à utiliser ce terme car il avait besoin de pouvoir utiliser quelque chose de plus court que *JavaScript+CSS[18]+DOM+XMLHttpRequest asynchrones* lors de discussions avec des clients.

Ajax est l'acronyme d'**A**synchronous **J**avaScript **A**nd **X**ML, soit en français JavaScript et XML asynchrones. Ajax n'est donc pas une nouvelle technologie en tant que telle mais désigne le fait que plusieurs technologies, développées séparément sont assemblées pour offrir de nouvelles possibilités et de nouvelles voies aux utilisateurs d'Internet.

Les principales technologies qui composent Ajax sont :

- Le JavaScript
- Le XML et le XSL[19]
- L'objet XMLHttpRequest
- Le XHTML[20] et les feuilles de styles CSS
- Le Modèle Object Document

Le JavaScript permet d'effectuer des traitements sur la page du client, avec l'aide du Modèle Objet Document (appelé par la suite DOM pour Document Object Model) qui permet quand à lui de parcourir et de manipuler un document HTML[21] ou XML au moyen de diverses méthodes.

Le XML a pour but de décrire et de structurer l'information tandis que le XSL est une manière de mettre en forme le contenu d'un document XML.

---

[17] [aja01]

[18] Glossaire – [C]

[19] Glossaire – [X]

[20] Glossaire – [H]

[21] Glossaire – [H]

Le XHTML et les feuilles de styles CSS permettent de définir la présentation des pages à l'utilisateur.

L'objet XMLHttpRequest va, quand à lui, lire des données contenues dans un fichier sur le serveur de manière asynchrone.

2.2 Principe de fonctionnement d'Ajax

Ajax fonctionne selon une architecture client/serveur, avec Ajax le serveur a pour but de retourner à l'objet XMLHttpRequest le contenu de la réponse à une requête, tandis que c'est le client qui va mettre en page ce contenu. Le protocole HTTP[22] est utilisé afin de transmettre les informations du client au serveur et vice-versa.

Ce qui rend Ajax particulier dans son utilisation par rapport à une application Web classique est un terme présent dans son nom : asynchrone. En effet, que se passe-t-il dans une application synchrone lors d'un appel au serveur ?

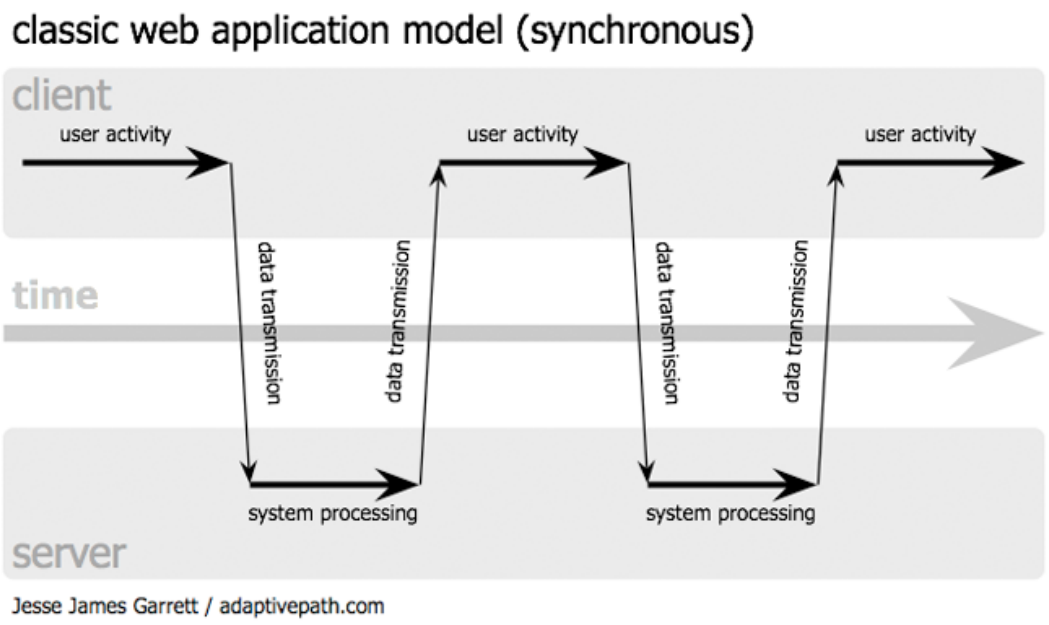

Figure 1 Schéma du fonctionnement d'une application Web classique

Comme le montre le schéma de la figure 1, à chaque fois qu'une action d'un utilisateur implique d'envoyer une requête au serveur, l'activité de l'utilisateur est interrompue le temps que le serveur traite et retourne les données qui lui ont été envoyées. L'utilisateur ne peut donc pas continuer à effectuer des actions et n'a aucune idée de ce qui se passe du côté serveur.

[22] Glossaire – [H]

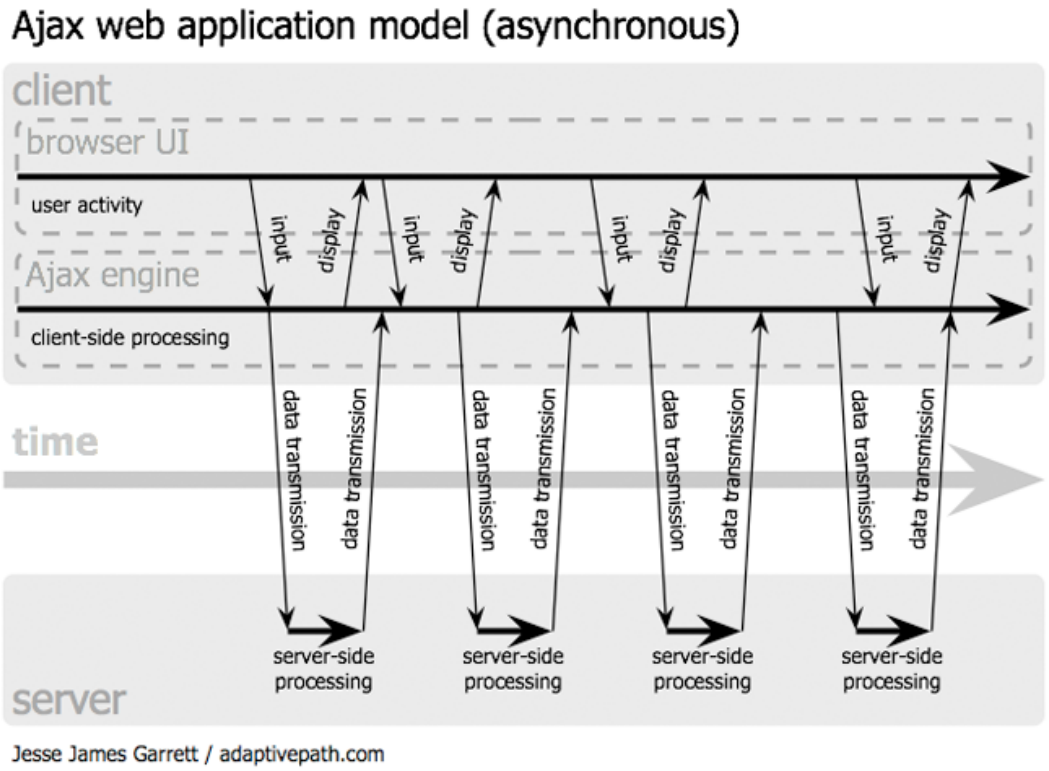

Figure 2 Schéma du fonctionnement d'une application Ajax

Dans le cas d'une application Web utilisant Ajax, l'activité de l'utilisateur n'est jamais interrompue. En effet lorsqu'une action de l'utilisateur requiert d'envoyer des données au serveur, JavaScript appelle le moteur Ajax qui va se charger de les envoyer de manière asynchrone permettant ainsi de ne pas interrompre l'interaction avec l'utilisateur qui ne voit ni la traditionnelle page blanche ni le sablier, présents lorsque le serveur travaille avec une application Web traditionnelle. L'utilisateur peut donc continuer de travailler ou de consulter la page pendant que le serveur traite la requête.

Comme le montre le schéma de la figure 2, le fait de faire appel au moteur Ajax ne veut pas obligatoirement dire que celui-ci va, immédiatement ou pas, envoyer une requête HTTP au serveur. Le moteur Ajax, qui connaît l'état de la requête, peut ainsi tenir au courant l'utilisateur de l'état de celle-ci.

Pour connaître l'état de la requête le moteur Ajax dispose d'une propriété qui va « réagir » lorsque l'état de la requête change. Ainsi le moteur est capable d'effectuer les traitements adéquats selon l'état de la requête.

2.3 L'apport d'Ajax dans les sites Web 2.0

Comme indiqué dans le paragraphe précédent, les applications Web synchrones ne sont pas vraiment ergonomiques pour l'utilisateur du fait que les nombreuses requêtes

du client au serveur entraînent une interruption des actions de l'utilisateur, empêchant ainsi une utilisation fluide de l'application pour l'utilisateur.

Ajax permet de remédier à ce problème, proposant ainsi des applications se rapprochant des applications de bureau où les requêtes sont transparentes pour l'utilisateur, sans que celui-ci n'ait eu à installer quoi que se soit.

Google, qui possède sur Internet de nombreuses applications utilisant Ajax, propose avec son service Google Maps[23] un exemple frappant de la mise en œuvre de cette nouvelle ergonomie. Google Maps est un service en ligne permettant de consulter une carte du monde en format plan ou vue satellite, ainsi que d'effectuer une recherche d'itinéraire.

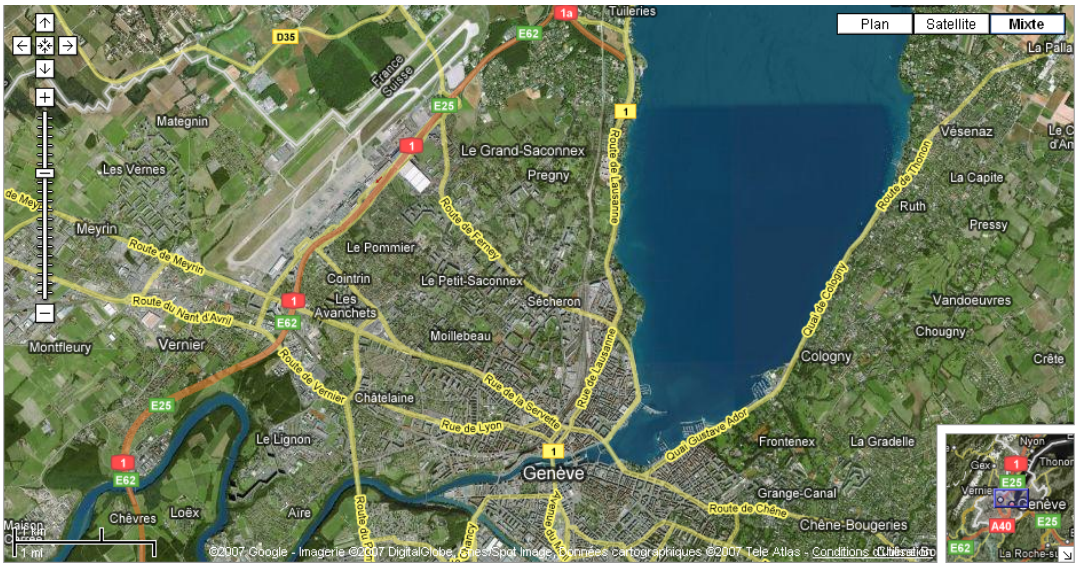

Illustration 1 Vue mixte avec Google Maps

Là où l'utilisation d'Ajax est frappante c'est lors de la consultation de la carte. En effet, lorsque l'on se déplace sur la carte, l'affichage lors du changement de zone est quasiment instantané. Pour ce faire il suffit à Google de détecter le mouvement de la souris et de calculer ainsi les prochaines zones à afficher, le tout se faisant sans aucun rechargement de la page.

Si cette application était conçue sans utiliser Ajax, chaque mouvement de la souris entraînerait le rechargement complet de la page, rendant ainsi l'utilisation de se service très pénible. Google ne propose d'ailleurs pas d'alternative dans le cas où Ajax serait inutilisable chez le client, ce service est tout simplement désactivé.

[23] http://maps.google.fr/

2.4 Le JavaScript

2.4.1 Historique du langage

JavaScript est apparu en septembre 1995 avec le navigateur Netscape 2.0, appelé tout d'abord Mocha, puis LiveScript lors de sa sortie, il prit son nom actuel, JavaScript, en décembre 1995.

Il y eut très vite un vif intérêt pour JavaScript tant chez les utilisateurs que chez les développeurs Internet. Microsoft, alors en retard dans le domaine de l'Internet, s'y intéressa rapidement et intégra JavaScript à son navigateur Internet Explorer 3 au milieu de l'année 1996.

Pour des raisons légales (Microsoft refusant d'acheter une licence à Netscape) et afin de pouvoir ajouter des fonctions supplémentaires à JavaScript, Microsoft appela JScript sa version de JavaScript. S'est alors posée la question de la compatibilité entre les navigateurs.

En 1997 Netscape et Microsoft choisirent l'organisme ECMA (pour **E**uropean **C**omputer **M**anufacture **A**ssociation) afin de standardiser le langage, ce qui sera fait avec la norme du nom d'ECMAScript, sous la référence ECMA-262[24]. Malheureusement cette standardisation n'a porté que sur la syntaxe de base et n'a de ce fait apporté qu'une compatibilité minimale. Un développeur peut donc être amené à développer un code différent mais produisant le même résultat pour chacun des navigateurs du marché.

Internet Explorer et Netscape ne sont pas les seuls navigateurs acceptant JavaScript ; parmi les plus célèbres nous pouvons citer Firefox, Safari, Opera ou encore Konqueror.

JavaScript est un langage de script incorporé aux balises[25] HTML et dont le code est géré et exécuté par le navigateur. JavaScript est donc exécuté du côté du client. Il faut noter que JavaScript peut aussi s'exécuter en dehors du navigateur et du côté serveur mais ce n'est pas ce qui nous intéresse avec Ajax. Enfin, malgré leur ressemblance au niveau du nom, JavaScript n'a rien à voir avec Java qui est un langage de programmation bien plus riche.

---

[24] http://www.ecma-international.org/publications/standards/Ecma-262.htm

[25] Glossaire – [B]

Du fait qu'il soit interprété par le navigateur le code JavaScript est visible à l'utilisateur (clic droit, afficher la source depuis le navigateur). Du point de vue de la sécurité, JavaScript ne peut ni lire ni écrire sur le disque dur d'un utilisateur en dehors de la zone réservée aux cookies[26]. De même JavaScript ne définit aucune instruction lui permettant de se connecter à un autre ordinateur ou à un serveur, en dehors de l'utilisation de l'objet XMLHttpRequest qui sera décrit plus bas.

JavaScript permet d'accroître l'interactivité avec l'utilisateur au moyen de scripts permettant par exemple de :

- Afficher l'adresse IP ou le navigateur employé par l'utilisateur.
- Rediriger l'utilisateur vers une autre page.
- Vérifier la saisie des champs de formulaires.
- Afficher des messages d'alerte, de confirmation ou d'erreur.
- Gérer les menus d'un site Internet.
- Animer du texte ou des images.
- Réagir à l'action de la souris.

Revers de la médaille, JavaScript peut être désactivé par l'utilisateur via le navigateur, il faut donc bien réfléchir à son utilisation.

2.4.2 Syntaxe du langage

2.4.2.1 Inclure du JavaScript

Il est possible d'écrire des scripts JavaScript en deux endroits d'une page HTML. On peut inclure du code JavaScript dans la balise <head> ou encore dans la balise <body> de la page HTML.

L'avantage que l'on peut retirer en incluant les scripts dans la balise <head> plutôt que dans <body> est que ces scripts seront disponibles au moment du chargement du contenu de la page HTML. En effet, celle-ci étant lue séquentiellement, les scripts écrits dans la balise <head> seront disponibles dès le chargement du contenu de la page.

Il y a deux manières de déclarer à la page HTML que l'on veut y ajouter du code JavaScript. La première est de l'écrire directement dans la page HTML comme indiqué ci-dessus. Cela se fait à l'aide de la balise <script> de la manière suivante :

[26] Glossaire – [C]

```
1  <script type="text/javascript">
2      //Le code JavaScript
3  </script>
```

L'indication type="text/javascript" indique au navigateur que l'on fait du JavaScript en mode texte. Cette indication est utile car il existe d'autres langages de script supportés par le HTML. Il est néanmoins possible de l'omettre, JavaScript étant reconnu comme langage par défaut par la majorité des navigateurs.

La seconde possibilité est d'écrire le code JavaScript dans un fichier qui sera inclus à la page HTML, cela se fait de la manière suivante:

```
5  <script type="text/javascript" src="script.js"></script>
```

2.4.2.2 Déclaration des variables

JavaScript est un langage à mots réservés. Il est faiblement typé. Une variable se définit soit à l'aide du mot réservé var (*var test = 123 ;*) soit implicitement par l'apparition d'un nom à gauche du signe d'assignation (*test = 123 ;*). Il est toutefois conseillé, pour faciliter la lecture du code, de prendre l'habitude de déclarer une variable grâce au mot réservé var. C'est ensuite l'interpréteur qui va se charger d'affecter à la variable le bon type en fonction du contexte. JavaScript n'oblige pas à affecter une valeur à une variable au moment de sa déclaration.

Le premier caractère du nom d'une variable doit impérativement commencer par une lettre, un underscore ou un dollar, tandis que les éventuels signes suivants devront être constitués de caractères alphanumériques, un underscore ou un dollar. Il est interdit de donner à une variable le nom d'un mot réservé. Il faut noter que JavaScript est sensible à la casse ce qui signifie que la variable *var test ;* n'est pas la même que la variable *var Test ;*

2.4.2.3 La séquence d'instructions

La séquence d'instructions :

- est une suite d'instructions élémentaires, terminées par le caractère ;
- peut contenir des variables locales ;
- est considérée dans sa globalité comme une instruction élémentaire.

De plus, les caractères { et } définissent respectivement le début et la fin des instructions de la séquence.

2.4.2.4 Les conditions

JavaScript permet de contrôler le flux d'exécution séquentielle à l'aide de deux types d'instructions conditionnelles : if et switch. Les instructions conditionnelles permettent d'exécuter une séquence plutôt qu'une autre lorsqu'une condition est vérifiée.

L'instruction if permet de tester une expression logique et d'exécuter une séquence d'instructions en fonction du résultat de l'évaluation. Voici la syntaxe de la condition if :

```
 7  if (condition 1){
 8      //Instruction 1;
 9  } else if (condition 2) {
10      //Instruction 2;
11      //Instruction 3;
12  } else {
13      //Instruction 4;
14  }
```

La deuxième structure conditionnelle implantée en JavaScript est l'instruction switch. Celle-ci permet de se brancher à une instruction donnée en fonction de la valeur d'une expression numérique entière ou d'une chaîne de caractères. En l'absence du mot réservé break l'exécution séquentielle se poursuit à partir du lieu de branchement. Voici sa syntaxe :

```
16  var variable = 0 ;
17  // divers traitements sur la variable
18  switch (variable){
19      case valeur 1:
20          //instruction 1;
21          break ;
22      case valeur 2:
23          //instruction 2;
24          break ;
25      case valeur 3:
26          //instruction 3;
27          break ;
28      default:
29          //instruction 4;
30  }
```

2.4.2.5 Les boucles

JavaScript intègre également quatre types de boucles : for, for…in, while et do…while.

Voici la syntaxe de la boucle for :

```
32  for (initialisation de la variable; condition de sortie; mise à jour de la variable){
33      //instruction 1;
34      //instruction 2;
35  }
```

La boucle for…in est une variante de la boucle for vue ci-dessus qui permet de spécifier la variable (par exemple un tableau) à utiliser lors du parcours de la boucle. Voici sa syntaxe :

```
37  for (valeur in tableau) {
38      //instruction 1;
39      //instruction 2;
40  }
```

Voici la syntaxe de la boucle while:

```
42  while (condition de confirmation) {
43      //instruction 1;
44      //instruction 2;
45      //mise à jour de la variable pour la condition de confirmation;
46  }
```

Il existe une variante de la boucle while, la boucle do…while, celle-ci a un fonctionnement similaire à la boucle while à la différence près que la condition de continuation est testée après que les instructions contenues dans la boucle aient été exécutées. Cela signifie que les instructions seront exécutées au minimum une fois, ce qui n'était pas le cas dans les autres types de boucles.

Voici sa syntaxe :

```
48  do {
49      //instruction 1;
50      //instruction 2;
51      //mise à jour de la variable pour la condition de confirmation;
52  } while (condition de confirmation);
```

De manière générale, il faut faire attention à ce que la séquence d'instructions sous contrôle de la boucle affecte bien la partie de l'état du système employé par l'expression conditionnelle, sans quoi il est possible de se retrouver en présence d'une boucle infinie.

2.4.2.6 Les procédures

Une procédure définit une suite d'instruction ne retournant pas de résultat. Elle est définie à l'aide du mot réservé *function* suivi du nom de la procédure puis des éventuels paramètres, séparés par une virgule, de la manière suivante :

```
54  function nomProcedure(paramètre1, paramètre2){
55      //suite d'instructions
56  } // nomProcedure
```

Dans le cas d'une procédure sans paramètre, la définition des paramètres de la procédure se fait à l'aide d'une paire de parenthèses vides.

Une procédure n'est pas exécutée lors de sa définition, pour l'exécuter il faut l'appeler de la manière suivante :

```
58  nomProcedure("valeurParamètre1", "valeurParamètre1");
```

Les paramètres ne sont visibles que dans la procédure, tout comme les variables créées localement à la procédure.

2.4.2.7 Les fonctions

Tout comme la procédure une fonction définit une suite d'instruction mais à comme différence le fait de retourner un résultat. Une fonction est définie de la même manière qu'une procédure, soit de cette manière :

```
60  function nomFonction(paramètre1, paramètre2){
61      //suite d'instructions
62      return résultat;
63  } // nomFonction
```

Le mot réservé *return* permet de retourner la valeur du résultat de la fonction.

Une fonction est exécutée en l'appelant de la manière suivante :

```
65  var résultatFonction = nomFonction("valeurParamètre1", "valeurParamètre1");
```

La valeur de retour de la fonction est ainsi affectée à la variable résultatFonction.

2.5 Le XML

2.5.1 Description générale

XML est l'acronyme d'e**X**tensible **M**arkup **L**angage. Comme le langage HTML, c'est un langage de balises mais contrairement à celui-ci, les balises du langage XML ne sont pas prédéfinies, XML est donc comme son nom l'indique, un langage extensible.

Alors que le but du langage HTML est d'afficher les données à travers des balises interprétées par le navigateur, XML à lui pour but de structurer l'information et il faudra au navigateur l'aide d'une feuille de style CSS ou d'un fichier XSL pour savoir comment interpréter les balises contenues dans un document XML.

XML est ce que l'on appelle un métalangage, soit un langage permettant de décrire d'autres langages.

2.5.2 Syntaxe du langage

Contrairement au langage HTML, XML est extrêmement strict sur la syntaxe à utiliser et la moindre erreur rend un document XML inutilisable[27]. Il est donc utile de préciser les principales règles syntaxiques qu'un document XML doit respecter.

Tout d'abord chaque balise ouverte doit être fermée et la dernière balise ouverte doit être la première à être fermée. Contrairement au langage HTML où le fait d'oublier de fermer certaines balises n'empêche pas le navigateur d'afficher correctement la page, XML ne tolère pas l'oubli de fermeture d'une balise. De même, l'inversion de l'ordre de fermeture d'une balise rend le document syntaxiquement incorrect.

Les balises peuvent contenir un ou plusieurs attributs. Ces attributs devront obligatoirement être entourés de guillemets.

Les noms des balises :

- peuvent contenir des caractères alphanumériques ainsi que des points des points, des doubles points, des traits d'union et des underscores ;
- ne peuvent commencer ni par un chiffre, ni par xml ;
- sont sensibles à la casse.

XML autorise les balises dites vides, elles doivent être fermées soit par <balise_vide /> soit par <balise_vide></balise_vide>.

Maintenant que les règles syntaxiques sont définies, voyons de quoi est constitué un document XML.

Un document XML doit toujours commencer par un prologue. Celui-ci doit contenir au minimum le numéro de version du langage. A l'heure actuelle la seule valeur possible pour cet attribut est "1.0". Il est également possible d'y spécifier le jeu de caractères utilisé à l'aide de l'attribut encoding.

Directement en dessous du prologue se trouve l'élément racine du document XML qui doit être unique et qui doit être fermé à la toute fin du document.

Seuls des commentaires peuvent se glisser entre le prologue et l'élément racine.

[27] Pour une description formelle de la syntaxe de XML, voir la BNF du langage en http://lists.xml.org/archives/xml-dev/199703/msg00083.html

Afin de mieux illustrer ces règles voici un petit document XML syntaxiquement correct.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- auteur: Alain Pellaux -->
<famille>
    <personne>
        <nom>Pellaux</nom>
        <prénom>Alain</prénom>
        <date_naissance>30 janvier 1984</date_naissance>
    </personne>
    <personne lien="soeur">
        <nom>Pellaux</nom>
        <prénom>Sophie</prénom>
        <date_naissance>30 juillet 1986</date_naissance>
    </personne>
    <personne lien="soeur">
        <nom>Pellaux</nom>
        <prénom>Marie</prénom>
        <date_naissance>24 août 1988</date_naissance>
    </personne>
</famille>
```

Figure 3 Document XML syntaxiquement correct

2.6 L'objet XMLHttpRequest

L'objet XMLHttpRequest est l'élément clé des applications Ajax. Il permet d'envoyer des requêtes HTTP au serveur de manière synchrone ou asynchrone.

2.6.1 Historique de la classe

XMLHttpRequest a tout d'abord été développé par Microsoft en tant que contrôle ActiveX[28] et est apparu avec Internet Explorer 5.0 en septembre 1998.

XMLHttpRequest a été intégré en temps qu'objet JavaScript natif[29] dans la première version du navigateur Mozilla, puis progressivement dans les principaux navigateurs du marché. XMLHttpRequest fut intégré comme objet JavaScript dans Internet Explorer à partir de la version 7 du navigateur.

2.6.2 Fonctionnement de la classe

Afin de pouvoir utiliser l'objet XMLHttpRequest, il faut tout d'abord créer une instance de l'objet, ce qui se fait d'une manière différente selon que l'on utilise Internet Explorer 6.0 ou inférieur ou un autre navigateur.

[28] Glossaire – [C]

[29] Glossaire – [N]

L'instance se créée de la manière suivante :

```
 9      var xhr = null;
10      // Pour Internet Explorer
11      if (window.ActiveXObject){
12          try{
13              xhr = new ActiveXObject("Microsoft.XMLHTTP");
14          }
15          catch (e){
16              xhr = new ActiveXObject("Msxml2.XMLHTTP");
17          }
18      // Autres navigateurs
19      } else if(window.XMLHttpRequest){
20          xhr = new XMLHttpRequest();
21          // ajoute un en-tête XML
22          if(xhr.overrideMimeType){
23              xhr.overrideMimeType("text/xml");
24          }
25      // Navigateur incompatible
26      } else {
27          alert("Navigateur incompatible avec AJAX");
28      }
29      return xhr;
```

Figure 4 Création d'une instance de XMLHttpRequest

Il faut tester si le navigateur du client supporte le contrôle ActiveX ou s'il supporte l'objet natif XMLHttpRequest afin de créer l'objet, appelé ici xhr, de la manière qui convient. Si le navigateur ne supporte pas l'objet XMLHttpRequest une alerte est affichée afin de le signaler à l'utilisateur.

2.6.3 Propriétés de la classe

Une fois l'objet XMLHttpRequest instancié, nous avons accès aux propriétés suivantes :

- readyState : contient l'état de l'objet pendant la requête, peut prendre 5 valeurs qui sont détaillées en 2.6.3.1.
- onreadystatechange : supporté en mode asynchrone uniquement, onreadystatechange se déclenche à chaque fois que readyState change d'état. Il est possible d'associer une fonction à onreadystatechange ce qui aura pour effet que la fonction sera appelée à chaque fois que readyState verra son état changer.
- status : contient le statut de la réponse du serveur, la liste complète des différents statuts et leur définition sont donnés sont disponibles sur Wikipedia[30].
- statusText : contient le texte correspondant au statut de la réponse du serveur.
- responseText : contient la réponse du serveur à la requête sous forme de texte.

[30] http://fr.wikipedia.org/wiki/Liste_des_codes_HTTP

- responseXML : contient la réponse du serveur à la requête sous forme de document XML.

2.6.3.1 Valeurs de readyState

Voici un tableau comprenant les valeurs, l'état et la description possible pour la propriété readyState.

Valeur	Etat	Description
0	unsent	La requête n'est pas initialisée.
1	open	L'invocation de la méthode open s'est correctement déroulée.
2	sent	La requête a bien été envoyé au serveur qui n'a pas encore retourné de données.
3	loading	Le serveur est en train de retourner les données.
4	done	Le transfert des données est terminé.

Tableau 1 Valeurs de readyState

2.6.4 Méthodes de la classe

Une fois l'objet XMLHttpRequest instancié, voici ses méthodes :

- open("méthode", "URL[31] du fichier", "mode", "identifiant", "mot de passe") : permet d'initialiser l'objet XMLHttpRequest pour une requête. La méthode indique si l'on veut faire une requête http en POST ou en GET. L'URL du fichier détermine l'adresse de la source. Mode est un booléen qui détermine si la requête est synchrone (false) ou asynchrone (true), ce paramètre est optionnel mais s'il n'est pas indiqué, la requête sera de type asynchrone. Identifiant et mot de passe sont quand à eux optionnels.
- send("données") : envoie la requête au serveur avec les paramètres spécifiés grâce à la méthode open. Si la méthode spécifiée est GET, données doit valoir "null", avec POST données doit être une chaîne de caractères du type "parametre1=valeur1&parametre2=valeur2".
- abort() : annule la requête.
- getAllResponseHeader() : ne peut être utilisé que lorsque readyState vaut 3 ou 4. Retourne tous les entêtes http sous la forme d'une chaîne de caractères.

[31] Glossaire – [U]

- getResponseHeader("entête") : ne peut être utilisé que lorsque readyState vaut 3 ou 4. Retourne la valeur de l'entête spécifié en paramètre.
- setResponseHeader("entête", "valeur") : ne peut être utilisé que lorsque readyState vaut 1. Permet d'assigner une valeur à un entête.
- overrideMimeType("mimeType") : oblige un document à être traité avec le mimeType spécifié en paramètre.

2.7 Le Modèle Objet Document

Le modèle objet document, plus connu sous l'acronyme de DOM (pour Document Object Model), permet d'accéder et de mettre à jour le style, la structure et le contenu d'un document HTML ou XML. Le DOM est standardisé depuis 1998 par le consortium W3C[32].

L'élément central du DOM est ce que l'on appelle un nœud, ou node en anglais. Ainsi tout élément d'un document est un nœud, ce qui donne la possibilité de représenter un document sous la forme d'un arbre. Le DOM permet d'ajouter, de modifier ou de supprimer des nœuds dans un document.

Chaque nœud possède des propriétés permettant de connaître :

- ses attributs (attributes).
- son nom (nodeName).
- son type (nodeType).
- sa valeur (nodeValue).

Les propriétés permettant de se déplacer dans le document à l'aide du DOM sont les suivantes :

- parentNode, qui permet de connaître le nom du nœud parent.
- childNodes, qui crée une liste de nœuds enfant.
- firstChild, qui permet de connaître le premier enfant d'une liste de childNodes.
- lastChild, qui permet de connaître le dernier enfant d'une liste de childNodes.
- nextSibling, pour un même nœud parent, permet de connaître le nom du nœud suivant, contient null si le nœud est le dernier.
- previousSibling, pour un même nœud parent, permet de connaître le nom du nœud précédent, contient null si le nœud est le premier.

La propriété childNodes est délicate à utiliser, en effet les navigateurs Mozilla prennent en compte les espaces comme des nœuds de type texte, ce que ne font pas les autres

[32] http://www.w3.org/DOM/

navigateurs. Ainsi des problèmes de compatibilité peuvent survenir si ce cas n'est pas traité.

Les diverses propriétés expliquées ci-dessus permettent d'accéder aux divers éléments d'un document. Il existe cependant d'autres méthodes qui permettent elles aussi d'accéder aux nœuds :

- getElementById() parcourt le document en recherchant le nœud dont l'identifiant a été passé en paramètre. L'identifiant doit être unique.
- getElementsByName() crée une liste d'éléments dont l'attribut name correspond à la valeur fournie en paramètre. La liste peut être parcourue de la même manière qu'un tableau.
- getElementsByTagName() crée une liste d'éléments dont le nom de la balise correspond à la valeur fournie en paramètre. La liste peut être parcourue de la même manière qu'un tableau.

Ces trois méthodes sont très utiles pour se rapprocher rapidement et simplement de l'élément que l'on souhaite atteindre. Il est donc fréquent d'utiliser ces méthodes pour se rapprocher de l'élément, puis les propriétés décrites plus haut afin d'accéder à l'élément lui-même.

2.8 Les avantages d'Ajax

Ajax permet plus de réactivité. Grâce à Ajax il n'y a plus besoin de recharger intégralement une page Web à chaque demande de l'utilisateur. Cela permet d'améliorer la réactivité, ainsi que l'ergonomie.

Ajax est portable. Les navigateurs récents supportant pour la plupart l'objet XMLHttpRequest, on peut dire qu'Ajax est portable.

Ajax fonctionne sur de simples postes clients. Ajax ne nécessite pas de grosses ressources mémoires et il n'y a pas de plugin[33] à ajouter pour pouvoir utiliser cette technologie.

2.9 Les défauts d'Ajax

Un site utilisant Ajax ne peut pas fonctionner si le JavaScript est désactivé. Les navigateurs permettant aux utilisateurs de désactiver le JavaScript, il faut alors soit prévoir une solution alternative, soit afficher un message demandant à l'utilisateur d'activer JavaScript afin d'employer le site Internet désiré.

[33] Glossaire – [P]

Ajax peut perturber le référencement d'un site. Les robots parcourant un site pour le référencer ne pouvant pas indexer le contenu construit dynamiquement, il existe donc le risque de n'être pas, ou peu référencé par les moteurs de recherche.

Ajax peut déstabiliser les utilisateurs. L'adresse URL d'une page ne changeant pas lorsque la page change dynamiquement, il est possible que les boutons précédent et suivant ne fonctionnent pas. De même qu'en mettant une page dans ses favoris Internet, l'utilisateur risque de se retrouver sur la page d'accueil plutôt que sur la page souhaitée. Il est néanmoins possible de résoudre ces problèmes.

Développer une application Ajax nécessite la connaissance de plusieurs langages, Ajax étant le regroupement de plusieurs technologies distinctes, le temps d'apprentissage pour réaliser une application Ajax est plus long et donc plus coûteux.

Les bugs sont difficiles à trouver. Ajax est principalement construit avec du JavaScript, qui est un langage connu pour être difficile à déboguer.

2.10 Les Frameworks

Un framework est une bibliothèque qui fournit un certain nombre de fonctions permettant d'accélérer le développement des applications. Ajax n'échappe pas à la règle et il y a un certain nombre de frameworks à disposition des développeurs. Aucun framework n'a été utilisé dans le cadre de ce travail de diplôme mais il nous semblait intéressant de signaler l'existence d'outils facilitant la tâche d'un développeur.

Dans un article du mensuel Programmez ![34], Gauthier DELAMARRE a classé les frameworks en 3 catégories :

- Les frameworks JavaScript
- Les frameworks d'interfaces graphiques
- Les frameworks Web 2.0 complets

2.10.1 Les frameworks JavaScript

Comme le nom l'indique, les frameworks JavaScript sont totalement écrits en JavaScript. Ils sont indépendants du langage utilisé pour la partie serveur et ils permettent de gérer tout ce qui concerne des appels à Ajax d'une application.

Parmi les avantages, certains frameworks JavaScript étendent le langage JavaScript et simplifient l'accès aux propriétés du DOM. De plus, la plupart de ces frameworks sont

[34] [fra01]

conçus de manière à être compatibles avec l'ensemble des navigateurs les plus répandus.

Le plus connu de ces frameworks est sans doute **Prototype JS**[35], qui étend le langage JavaScript et simplifie la manipulation des objets et propriétés du DOM. Il possède de plus une documentation complète.

2.10.2 Les frameworks d'interfaces graphiques

Les frameworks d'interfaces graphiques proposent une solution simple pour améliorer le côté graphique d'une application Web et ainsi se rapprocher des applications écrites en Flash.

Les plus connus de ce type de frameworks sont tout d'abord **scriptaculous**[36] qui offre des possibilités tels que le glisser/déposer, divers effets sur les textes, des calques HTML ou du morphing. Il ne dispose par contre pas d'une documentation à la hauteur de ses capacités. Il existe aussi **Dojo**[37] qui offre de nombreuses possibilités aux développeurs mais qui est néanmoins lourd au démarrage.

2.10.3 Les frameworks Web 2.0 complets

Les frameworks Web 2.0 complets sont des frameworks qui cumulent les possibilités offertes par les deux styles de frameworks précités. Ils permettent ainsi de réaliser entièrement un projet.

Le plus connu des ces frameworks est sûrement **Ruby on Rails**[38]. Apportant de la rigueur et implantant des designs patterns tel que le Modèle-Vue-Contrôleur (MVC), Ruby on Rails à l'avantage de posséder une grande communauté, qui propose de nombreux plugins pour le framework.

Google a lui aussi son framework nommé **Google Web Toolkit**[39]. Google Maps ou Gmail ont ainsi été créés à l'aide de ce framework.

---

[35] http://www.prototypejs.org

[36] http://script.aculo.us

[37] http://www.dojotoolkit.org

[38] http://www.rubyonrails.com

[39] http://code.google.com/webtoolkit

2.11 Les alternatives

Ajax n'est pas le seul moyen d'obtenir une application Web proche de celles de bureau et ce chapitre présente les technologies pouvant concurrencer Ajax. L'objectif n'est cependant pas de faire une comparaison avec Ajax en terme de qualité ou de facilité de mise en œuvre, ce qui impliquerait de connaître et d'avoir testé ces solutions, mais simplement de montrer qu'il existe des alternatives à Ajax.

2.11.1 Les frames

Les frames sont des sous fenêtres qui divisent la zone d'affichage d'un navigateur. Les frames sont contenues dans un frameset qui est le fichier chargé par le navigateur. Celui-ci contient diverses informations tels que le nom, la taille ou la position de chaque frame, mais pas leur contenu. Chaque frame peut recevoir du contenu indépendamment des autres et il existe la possibilité de modifier la visibilité d'une frame.

Alternativement aux frames, il existe les iframes, qui sont des sections de fenêtres indépendantes contenues dans une page Web et qui permettent de recharger un fichier dans un iframe sans modifier les autres sections d'une page.

2.11.2 Flex

Propriété d'Adobe, Flex[40] est une solution de développement basée sur Flash pour la présentation et utilisant MXML et ActionScript 3.0 pour la programmation. Flex est utilisable sur tout poste client ayant installé le plugin Flash.

Disposant d'outils de développement performants, Flex permet d'obtenir des applications Web réactives, comme le démontre Adobe avec l'exemple de l'Hybrid Store[41].

2.11.3 Les autres

Adobe n'est pas seul à occuper ce segment de marché. Microsoft propose, pour concurrencer Flex, son propre plugin du nom de Silverlight[42], alors que Sun propose JavaFX[43].

---

[40] http://www.adobe.com/fr/products/flex/

[41] http://examples.adobe.com/flex2/inproduct/sdk/hybridstore/hybridstore.html

[42] http://www.microsoft.com/silverlight/default_ns.aspx

3. Prototype

Afin de mettre en œuvre les concepts étudiés tout au long de ce document, nous avons décidé de développer un prototype d'application utilisant Ajax. Celui-ci est disponible sur le site du campus de la HEG à l'adresse suivante : http://campus.hesge.ch/asap/commune.html.

Le prototype a été testé et est fonctionnel sur les navigateurs suivants :

- Mozilla Firefox 2.0
- Internet Explorer 7
- Safari 3 pour Windows
- Opera 9

Il n'a pu être testé sous Konqueror faute de distribution Linux disponible.

3.1 Description du Prototype

Le choix du prototype à développer dans le cadre de ce travail de diplôme s'est fait en prenant en compte différentes contraintes, il était en effet important que le prototype :

- montre bien la différence entre une application Web « classique » et une application Web Ajax ;
- soit compatible avec les principaux navigateurs du marché ;
- ne soit pas un trop gros projet compte tenu du temps à disposition pour la réalisation du travail de diplôme ;
- n'utilise pas de framework afin de pouvoir mieux étudier le comportement des différentes technologies composant Ajax.

A partir de ces différentes contraintes, nous avons décidé de réaliser un prototype affichant des données correspondantes à la saisie d'un utilisateur, car en plus de respecter les contraintes énoncées ci-dessus, ce prototype nous a permis de nous concentrer sur l'aspect technique d'Ajax, la partie graphique étant simplement l'affichage des données dans un tableau.

Le prototype s'intitule liste des communes de Suisse romande et, comme son nom l'indique, permet d'afficher la liste des communes des cantons de Genève, Fribourg, Neuchâtel, Jura, Vaud et Valais. En plus du nom de la commune, la localité et son numéro d'acheminement postal, les armoiries du canton auquel la commune appartient sont également affichées à l'écran.

---

[43] http://www.sun.com/software/javafx/

Illustration 2 Résultat d'une saisie

3.2 Conception du prototype

Avant d'expliquer le fonctionnement du prototype, voici les différentes étapes de la conception du prototype.

Tout d'abord il a fallu rechercher une liste des communes de Suisse romande qui fut trouvée sur le site http://www.admin.ch[44]. Il a ensuite fallu transformer le fichier Excel en un fichier plat contenant uniquement la liste des communes romandes ainsi que les informations nécessaires.

Ces informations devant figurer dans une base de données il était dès lors facile de transformer les informations du fichier plat en un script de base de données grâce à un script PHP[45] créé pour l'occasion. Il ne sera pas fait mention de ce script dans la suite du document car il n'influence pas le fonctionnement même du prototype.

Il a ensuite fallu créer la partie serveur du prototype, puis la page de saisie et d'affichage des résultats et enfin le moteur Ajax du prototype. Le fonctionnement de ces pages est expliqué dans le paragraphe suivant.

3.3 Fonctionnement du prototype

Les différents fichiers du prototype sont listés dans l'annexe 2.

[44] http://www.bfs.admin.ch/bfs/portal/fr/index/infothek/nomenklaturen/blank/blank/e_c/02.html

[45] voir le fichier conv_csv.php, inclus dans le fichier annexe.pdf fourni en annexe

3.3.1 Rôle des fichiers

Comme le schéma ci-dessous le montre, le prototype est composé de quatre fichiers.

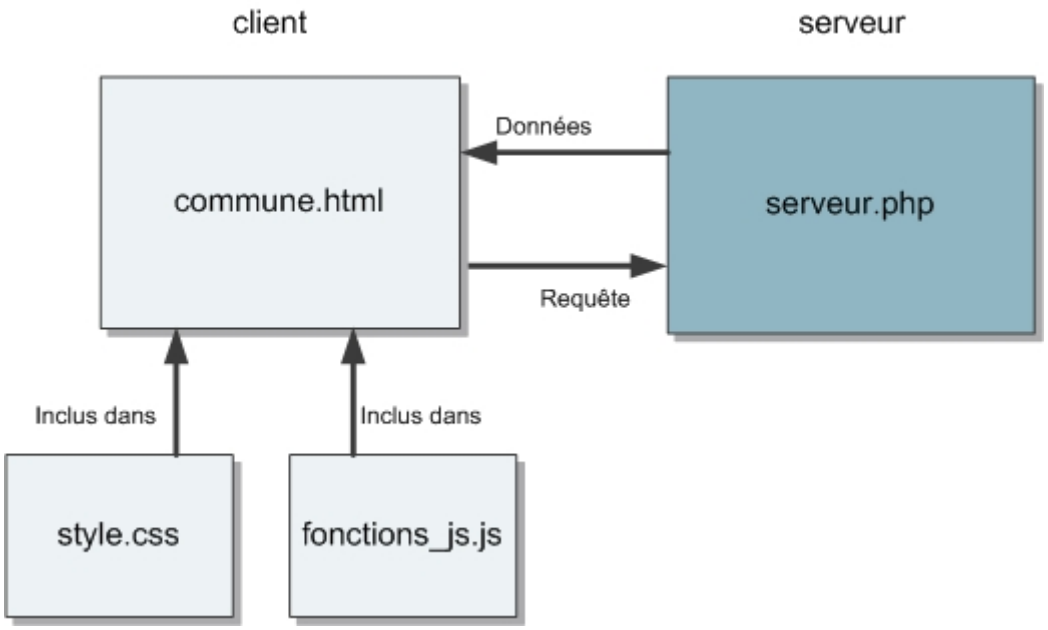

Figure 5 Liste des fichiers du prototype

Les fichiers en gris sont du côté client et sont, dans le cas d'une application Web, exécuté par le navigateur de l'utilisateur tandis que le fichier en vert est exécuté du côté du serveur.

L'utilisateur ne voit que la page ***commune.html***. C'est en effet elle qui a pour but d'afficher le formulaire de saisie ainsi que la liste des communes correspondantes à la dite saisie.

Le fichier ***style.css*** est la feuille de style du prototype, soit le fichier permettant de mettre en page les différents éléments. Il aurait été possible de mettre les éléments contenus dans ce fichier directement dans ***commune.html*** mais le fait de créer un fichier CSS externe permet de pouvoir offrir aux utilisateurs différentes mises en forme de la page très simplement. Il existe toutefois un bémol à ce point, dans le sens ou un tableau de tableau a été utilisé pour l'affichage des résultats, ce qui limite la modification de mise en page à travers le fichier ***style.css*** uniquement.

Le fichier ***fonctions_js.js*** contient lui toutes les fonctions nécessaires au fonctionnement d'une application Ajax. Comme pour la feuille de style il aurait été possible d'inclure les fonctions directement dans le fichier ***commune.html*** mais le fait de séparer les traitements de l'affichage permet une meilleure réutilisation du code, dans le cadre d'une autre application Ajax par exemple.

Le fichier ***serveur.php***, totalement invisible pour l'utilisateur, est lui chargé de retourner les données correspondant à la saisie de l'utilisateur au client sous la forme d'un document XML construit dynamiquement. Pour ce faire, le fichier va chercher les données correspondant à la saisie dans une base de données.

3.3.2 Technologies utilisées

Le prototype utilise les technologies suivantes :

- Le JavaScript
- Le XML
- L'objet XMLHttpRequest
- Les feuilles de styles CSS
- Le Modèle Object Document

Afin de mieux comprendre le fonctionnement du prototype voici un schéma montrant les interactions entre les différents processus de l'application.

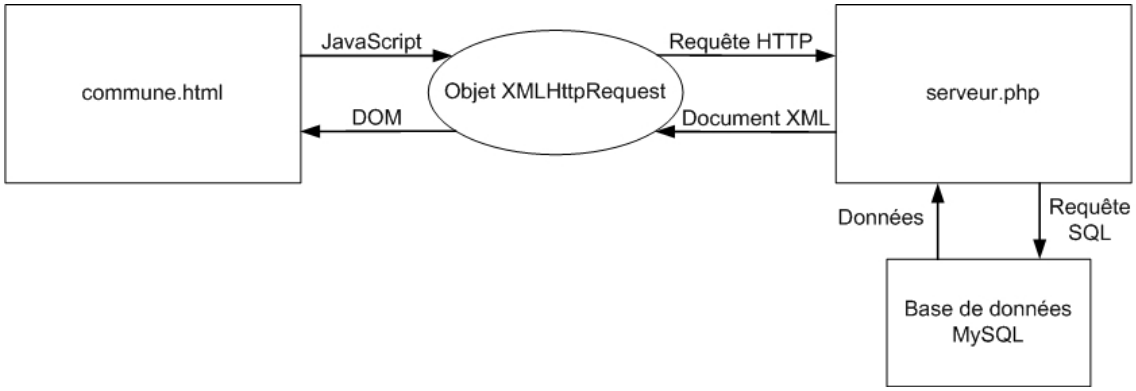

Figure 6 Interaction entre les processus de l'application

Le schéma de la figure 6 décrit les différentes étapes découlant d'une saisie de l'utilisateur. Ainsi pour chaque saisie de l'utilisateur, JavaScript est appelé afin de lancer le moteur Ajax. Celui-ci va alors créer une instance de l'objet XMLHttpRequest. L'instance aura la tâche d'envoyer la requête au fichier ***serveur.php*** qui, comme son nom l'indique, joue le rôle du serveur, se chargeant d'envoyer une requête à la base de données stockant les informations sur les communes.

Une fois que la base de données a retourné la liste des résultats correspondant à la requête, le serveur va, à partir de ces résultats, générer un document XML qui sera retourné à notre objet XMLHttpRequest. A partir de ce document le DOM permet de récupérer les données et de les afficher dans le document HTML, à savoir la page ***commune.html*** de départ.

Le tout se fait évidemment dans la plus grande transparence pour l'utilisateur, lequel ne voit aucun rechargement de page.

3.3.3 Par rapport à une application synchrone

La première différence avec une application synchrone est visuelle, le formulaire de saisie ne contient pas de bouton de validation. Dans ce cas l'utilisateur n'a pas besoin de valider sa demande, ceci étant fait automatiquement, améliorant ainsi l'ergonomie de l'application.

De plus l'action de l'utilisateur n'est jamais interrompue, ainsi si celui-ci venait à faire une faute de frappe, il n'a pas à attendre que la requête ait abouti pour modifier sa saisie.

3.3.4 Problèmes rencontrés

Créer une application amène souvent son lot de problèmes et ce prototype n'a pas échappé à la règle.

Le premier problème rencontré concernait l'envoi du document XML par le serveur à l'objet XMLHttpRequest. Le XML étant un langage très strict on aurait pu penser que si le serveur affichait correctement le document XML dans le navigateur, cela voulait dire que le document XML était valide, mais ce n'est pas le cas. Si le navigateur affichait bien le document XML demandé, la propriété responseXML contenait un message d'erreur indiquant que le document XML était mal formé. Le problème a été résolu en modifiant, dans le fichier ***serveur.php***, les guillemets simples et guillemets doubles.

Le deuxième problème rencontré concernait l'encodage des caractères. En effet lorsque la saisie contenait des accents (par exemple Céligny), le prototype fonctionnait correctement avec Firefox 2.0 mais pas avec Internet Explorer 7, qui ne retournait aucun résultat. Le problème provenait de l'encodage de la saisie lors de l'envoi de la saisie par le formulaire à la fonction qui traite l'objet XMLHttpRequest. Cela a été résolu en utilisant la fonction JavaScript encodeURIComponent lors de l'envoi de la saisie à la fonction, qui a pour but de crypter l'URI[46] en transformant les caractères spéciaux en séquence de signes ASCII[47].

[46] Glossaire – [U]

[47] Glossaire – [A]

Enfin le dernier problème concernait la gestion du copier-coller à l'aide de la souris uniquement. En effet l'événement JavaScript utilisé pour appeler le moteur Ajax est onKeyUp, qui détecte le relâchement d'une touche du clavier. Le copier-coller à l'aide du clavier est ainsi détecté puisqu'une touche est relachée alors que la souris passe au travers de cet événement. Le problème vient ici des événements disponible en JavaScript, puisqu'il n'existe pas d'événement se déclenchant lorsque le champ de saisie change de valeur (il existe bien un événement onChange mais il ne se déclenche que lorsque le champ perd le focus). Ce problème a été partiellement résolu. Pour ce faire une méthode qui détecte le clic droit de la souris a été utilisée ; une fois le clic droit effectué, la méthode, à l'aide de la fonction setTimeout, lance le moteur Ajax deux secondes après le clic droit. L'utilisateur dispose donc de deux secondes pour effectuer son copier-coller à la souris, passé ce délai le copier-coller ne sera pas détecté.

3.3.5 Améliorations possibles

Bien que ce prototype soit tout à fait opérationnel, il serait néanmoins possible de l'améliorer.

Tout d'abord il serait possible de faire une meilleure prise en charge du copier-coller à la souris (problème expliqué au chapitre 3.2.4), en lançant par exemple le moteur Ajax tout les xx milli secondes après un clic droit. Le problème est néanmoins plus complexe qu'il n'y paraît puisqu'il faudrait prévoir l'arrêt du moteur en cas d'autre événement appelé.

Il serait également possible de mettre en page les résultats sans utiliser de tableau comme c'est le cas dans cette version du prototype, par exemple en utilisant les balises couplées avec les feuilles de style CSS, ce qui pourrait sensiblement améliorer les performances du prototype.

Cette version du prototype se contente d'afficher une alerte si le navigateur ne supporte pas l'utilisation de l'objet XMLHttpRequest ou un message si le JavaScript n'est pas activé. Or il aurait été possible de faire une version fonctionnant à la manière d'une application Web synchrone parallèlement à la version fonctionnant avec Ajax.

Le prototype appelle le moteur Ajax à chaque touche du clavier relâchée par l'utilisateur. Il aurait été possible de créer un système de cache stockant les dernières saisies permettant ainsi d'afficher les résultats correspondant sans devoir rappeler le moteur Ajax, améliorant ainsi les performances du prototype.

Concernant ces propositions d'améliorations Il faut noter qu'elles n'impliqueraient pas une modification du code Ajax du prototype et qu'en ce sens elles sortent quelque peu du cadre de ce travail diplôme.

3.4 Code du prototype et explications

Le code complet des différents fichiers composants le prototype est fourni dans l'annexe 2. Ce chapitre a pour but d'expliquer les parties du code non triviales et qui n'ont pas été commentées dans le fichier même.

3.4.1 commune.html

Lignes 6 et 7 : ces deux lignes servent à inclure au fichier commune.html les fonctions JavaScript ainsi que la feuille de style CSS

Ligne 10 : l'utilisateur ne verra le message indiqué dans la balise uniquement si son navigateur ne peut pas exécuter le code compris dans la balise <script>. C'est le cas ici lorsque JavaScript est désactivé.

Ligne 13 : la fonction clic_droit est appelé lorsque l'utilisateur appuie sur le bouton de la souris.

Ligne 15 : l'événement onKeyUp appelle la fonction go() à chaque fois qu'une touche du clavier est relâchée, en passant en paramètre à la méthode la saisie de l'utilisateur. La méthode encodeURIComponent transforme les caractères spéciaux en signe ASCII.

Ligne 16 : permet d'afficher l'état de la requête au serveur (en cours de traitement ou ok).

Ligne 19 : permet d'afficher les résultats de la requête.

3.4.2 fonctions_js.js

Lignes 47 à 49 : lors de la mise en page permet d'aller à la ligne toutes les 5 colonnes du tableau. Le signe % signifie modulo en JavaScript.

Lignes 70, 74, 75, 77 et 78 : la propriété innerHTML permet d'insérer du contenu dans une balise HTML, ici la balise concernée est indiquée à l'aide la propriété getElementById.

3.4.3 serveur.php

Ligne 22 : c'est une syntaxe simplifiée d'un si…sinon, si la variable $_GET['saisie'] est définie elle est affectée à $saisie, sinon $saisie vaut ''.

3.5 Outils utilisés pour le développement

Pour la réalisation de ce prototype, les outils utilisés pour le développement ou le déploiement sont tous des outils gratuits, sauf editplus mais celui-ci est disponible à la HEG.

3.5.1 Editeur de texte

L'éditeur de texte utilisé pour le développement est editplus 2.31. Compatible HTML, JavaScript, PHP, CSS et d'autres, il est très simple d'utilisation et permet le coloriage syntaxique. Il est à noter que n'importe quel éditeur de texte aurait pu faire l'affaire.

3.5.2 Déploiement en local

Afin de tester l'application en local, nous avons utilisé easyPHP 1.8. Cet environnement de travail était nécessaire car étant donné que le prototype utilise un fichier PHP comme serveur ainsi qu'une base de données, il fallait pouvoir exécuter ses composants. EasyPHP permet donc d'exécuter du PHP à travers un serveur Apache et d'utiliser une base de données MySQL à l'aide de PHPMyAdmin.

3.5.3 Déboguage

Comme déjà indiqué, le JavaScript est un langage difficile à déboguer. Il existe néanmoins deux outils de développement Web très pratiques qui sont des plugins du navigateur Firefox.

Web Developer 1.1.4 permet, de détecter les erreurs JavaScript et CSS. Firebug 1.05 est lui plus complet puisqu'en plus de détecter les erreurs JavaScript et CSS, il permet aussi de vérifier le code HTML d'une page ainsi que de surveiller les requêtes de l'objet XMLHttpRequest.

3.5.4 Mise en ligne

Afin de mettre en ligne le prototype sur campus, le client FTP FilleZilla 2.2.32 a été utilisé.

Conclusion

Les sites Internet Web 2.0 se sont multipliés ces dernières années et se rapprochent de plus en plus des applications de bureau.

Ajax a permis ce rapprochement grâce notamment à son fonctionnement asynchrone lui permettant de ne recharger que la partie de la page qui a été modifiée, de manière transparente, améliorant ainsi l'ergonomie d'un site Internet.

Certains sites ne pourraient tout simplement pas exister sans ce fonctionnement asynchrone, nous pensons plus particulièrement à Google Maps dont le rechargement complet de la page à chaque mouvement de la carte rendrait le site inutilisable.

Ajax a de plus l'avantage de fonctionner sans devoir installer de plugin et les navigateurs supportent les technologies composant Ajax.

Ajax n'est pourtant pas exempt de défauts, à commencer par le fait qu'il ne peut fonctionner qu'avec les navigateurs dont l'utilisateur n'a pas désactivé JavaScript, de même que le programmeur doit prévoir une application compatible avec les principaux navigateurs du marché, le code d'une application pouvant être légèrement différent suivant le navigateur utilisé.

De plus, lors de la réalisation de l'application, nous nous sommes rendu compte que JavaScript n'est pas un langage facile à déboguer. Il existe heureusement des plugins pour le navigateur Firefox permettant de trouver les bugs plus facilement. Il existe aussi divers frameworks Ajax, aidant le développeur à créer une application.

D'un point de vue personnel, nous pensons qu'Ajax est un bon moyen de rapprocher les applications Web, de celles de bureau et pourquoi pas de les remplacer dans le futur. Ce n'est d'ailleurs pas un hasard si des acteurs majeurs de l'informatique tels que Windows ou Adobe, par exemple, commencent à mettre à disposition des internautes des applications Web permettant de produire des résultats identiques à leur cousines de bureau.

Comme nous l'avons vu dans ce document, Ajax permet de ne recharger que la partie de la page Web ayant été modifiée, ce qui permet d'économiser de la bande passante. A l'heure où les spécialistes prédisent qu'Internet pourrait saturer aux environs de 2010, Ajax est un moyen intéressant pour créer des applications Web moins gourmandes en bande passante, permettant ainsi de désengorger le réseau Internet.

De manière générale, ce travail de diplôme nous a permis d'acquérir les bases nécessaires à la réalisation d'une application Web 2.0 avec Ajax.

Nous avons comme regret le fait de ne pas avoir pu comparer Ajax avec les technologies concurrentes tel que Flex. Cela aurait été intéressant de pouvoir effectuer cette étude comparative, mais il n'était néanmoins pas possible d'apprendre à maîtriser correctement plusieurs technologies dans le temps imparti. Nous nous sommes donc concentré sur celle qui nous semblait la plus intéressante.

Bibliographie

[ade01] SUREAU, Denis. Tutoriel AJAX (Asynchronous JavaScript And XML) [en ligne]. http://www.xul.fr/xml-Ajax.html (consulté le 17.09.2007).

[ade02] WIKIPEDIA. Asynchronous JavaScript and XML [en ligne] http://fr.wikipedia.org/wiki/AJAX (consulté le 17.09.2007).

[ade03] VAN LANCKER, Luc, Présentation générale d'Ajax. In : Ajax, Développez pour le Web 2.0. France : eni éditions, 2007. 498 p.

[aja01] GARRETT, Jesse James. Ajax : A New Approach to Web Applications. In : Adaptive Path [en ligne]. Publié le 18 février 2005. http://www.adaptivepath.com/ideas/essays/archives/000385.php (consulté le 17.09.2007).

[aja02] GARRETT, Jesse James. Ajax : Questions et Réponses. In : Scriptet [en ligne]. Publié le 26 octobre 2006 http://www.scriptet.net/Ajax-faq.html (consulté le 17.09.2007).

[alt01] STEYER, Ralph. Alternatives à Ajax. In :Ajax et PHP. Marsat : CampusPress, 2006. 312 p.

[alt02] WIKIPEDIA. Adoble Flex [en ligne] http://fr.wikipedia.org/wiki/Adobe_Flex (consulté le 08.11.2007).

[dom01] VAN LANCKER, Luc, Le DOM (Document Object Model). In : Ajax, Développez pour le Web 2.0. France : eni éditions, 2007. 498 p.

[dom02] TEMPLIER, Thierry et GOUGEON, Arnaud. Programmation DOM. In : JavaScript pour le Web 2.0. Jouve : Eyrolles, 2007. 492 p.

[fra01] DELAMARRE, Gauthier. Comment choisir son framework Ajax ? Programmez !, juillet/août 2007, no 99, p. 36-38.

[js01] WERZ, Christian. Les bases de JavaScript. In : JavaScript, l'essentiel du code et des commandes. Marsat : CampusPress, 2007. 267 p.

[js02] VAN LANCKER, Luc. Le JavaScript. In : AJAX, Développez pour le Web 2.0. France : eni éditions, 2007. 498 p.

[js03] TEMPLIER, Thierry et GOUGEON, Arnaud. Fondements de JavaScript. In : JavaScript pour le Web 2.0. Jouve : Eyrolles, 2007. 492 p.

[web01] GONZALEZ, Dominique. Systèmes et réseaux. In : Groupe de recherche sur l'apprentissage automatique [en ligne]. Publié le 24.08.2005. http://www.grappa.univ-lille3.fr/polys/systreseaux/index.html (consulté le 26.09.2007).

[web02] GRUYER, Vincent. Tutorial, tuteuriel web : proposition de définition pour le Web 1.0, Web 2.0 et Web 3.0. In : Site photos panographiques [en ligne]. Publié le 07.05.2007. http://www.photos-panographiques.com/article-10301593-6.html (consulté le 19.09.2007).

[web03] MARKOFF, John. Entrepreneurs See a Web Guided by Common Sense. In : The New York Times [en ligne]. Publié le 12 novembre 2006.

http://www.nytimes.com/2006/11/12/business/12web.html?ex=1164171600&en=3a182a17940e3901&ei=5070 (consulté le 25.10.2007).

[web04] O'REILLY, Tim. What is Web 2.0. In : O'Reilly [en ligne]. Publié le 30 septembre 2005. http://www.oreilly.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html (consulté le 24.09.2007).

[xhr01] TEMPLIER, Thierry et GOUGEON, Arnaud. Mise en œuvre d'Ajax. In : JavaScript pour le Web 2.0. Jouve : Eyrolles, 2007. 492 p.

[xhr02] VAN LANCKER, Luc, L'objet XMLHttpRequest. In : AJAX, Développez pour le Web 2.0. France : eni éditions, 2007. 498 p.

[xhr03] SUREAU, Denis. L'Objet XMLHttpRequest. In : L'Objet XMLHttpRequest [en ligne]. Publié le 18.06.2007. http://www.xul.fr/XMLHttpRequest.html (consulté le 04.10.2007).

[xml01] VAN LANCKER, Luc, Introduction au XML. In : AJAX, Développez pour le Web 2.0. France : eni éditions, 2007. 498 p.

Annexe 1. Glossaire

A

- Acronyme : mot formé avec la première lettre de plusieurs mots.

- ASCII : **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange, norme de codage des caractères.

- ASP : **A**ctives **S**erver **P**ages, langage de programmation exécuté du côté serveur et créé par Microsoft.

B

- Balise : série de caractères entourant une phrase et servant à la définir et/ou à la mettre en forme.

- Blog : journal tenu par une ou plusieurs personnes dont les articles sont publiés sous forme de billets, du plus récent au plus ancien.

- Brainstorming : méthode de réflexion collective et quantitative permettant de sortir un grand nombre d'idées générales en peu de temps.

C

- Contrôle ActiveX : inventé par Microsoft, peut être téléchargé et exécuté par un navigateur ce qui lui permet d'accéder aux élément d'un environnement Microsoft.

- Cookies : fichier écrit sur le poste client par le serveur contenant des informations personnelles tels qu'un identifiant ou un mot de passe.

- CSS : **C**ascading **S**tyle **S**heet, langage permettant de gérer la présentation d'un site Web.

F

- Flux RSS : **R**eally **S**imple **S**yndication, flux qui, une fois que l'on s'y est inscrit, permet d'être tenu au courant des mises à jour d'un site sans devoir consulter celui-ci.

H

- HTML : **H**yper **T**ext **M**arkup **L**angage, langage de balisage hypertexte permettant d'écrire des pages Web statiques.

- HTTP : **H**yper **T**ext **T**ransport **P**rotocol, protocole permettant de transférer des fichiers entre un serveur et un client.

- Hypertexte : système permettant de passer d'un document lié à un autre.

M

- Module : Complément d'une application, permet de rajouter des fonctionnalités à celui-ci.

N

- Natif : caractérise quelque chose destiné à l'origine à une plate-forme fixée.

P

- PHP : **P**re **H**ypertext **P**rocessor, langage de programmation Internet permettant de produire des pages dynamiques.

- Plugin : programme permettant d'apporter de nouvelles fonctionnalités à un logiciel.

U

- URI : **U**niform **R**esource **I**dentifier, référence vers une ressource abstraite ou physique.

- URL : **U**niform **R**esource **L**ocator, adresse permettant d'accéder à un site Internet.

X

- XHTML : e**X**tensible **H**yper **T**ext **M**arkup **L**angage, successeur du HTML.

- XSL : eXtensible Stylesheet Langage, langage permettant de définir des feuilles de styles.